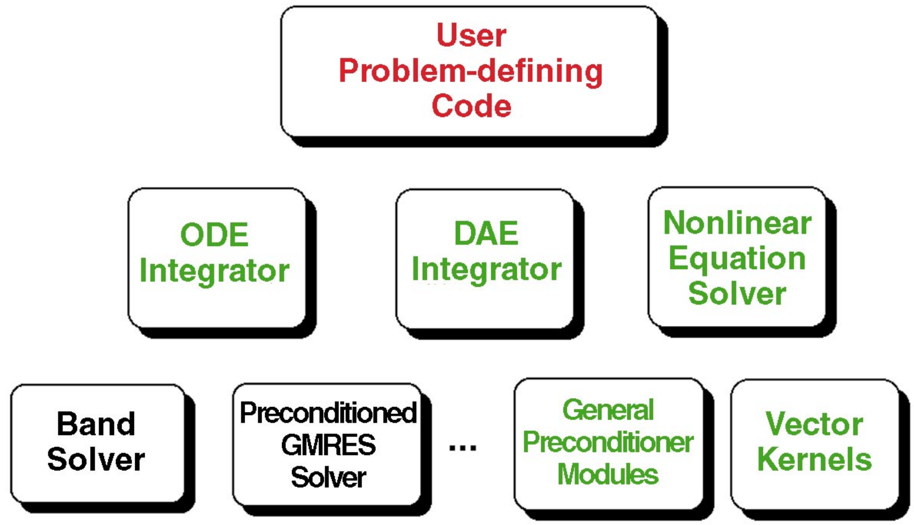
University of California
Lawrence Livermore
National Laboratory

PVODE, a Parallel Solver for Ordinary Differential Equations



The integrator PVODE, as well as related solvers for nonlinear systems and differential–algebraic systems, use a modular and extensible design.

Technology

PVODE is a portable solver for ordinary differential equation systems, based on robust mathematical algorithms, and targeted at large systems on parallel machines. Closely related software will also be developed for systems of nonlinear algebraic equations and for systems of differential–algebraic equations.

Applications

We are working with application efforts to use these parallel solvers, beginning with the tokamak edge-plasma model UEDGE.

A great many modeling situations lead to systems of ordinary differential equations, systems of nonlinear algebraic equations, or systems of differential–algebraic equations. As application codes require higher resolution and / or greater speed, they must move to parallel processors, where they will need parallel versions of such solvers.

The size and complexity of these problems pose major challenges. Time-dependent problems generally require implicit integration methods, which require the solution of nonlinear algebraic systems. In all cases, the nonlinear system is treated by a Newton iteration, which requires the solution of linear systems, which are typically large and sparse. For such systems, iterative (Krylov subspace) methods are an attractive choice, but to be robust, these require preconditioning.

General-purpose (sequential) software packages developed at LLNL for these three problem classes are among the most widely used solvers anywhere for these problems. Of central interest here is a C-language solver called CVODE (Scott D. Cohen and Alan C. Hindmarsh, "CVODE, a Stiff / Nonstiff ODE Solver in C," Computers in Physics, **10** (2), March / April 1996). Its highly modular structure was designed with parallel extensions in mind.

Modular Approach

In this effort, we are building a code system for parallel machines that will include three solvers. We adopt the Single Program, Multiple Data programming model, with message-passing communication.

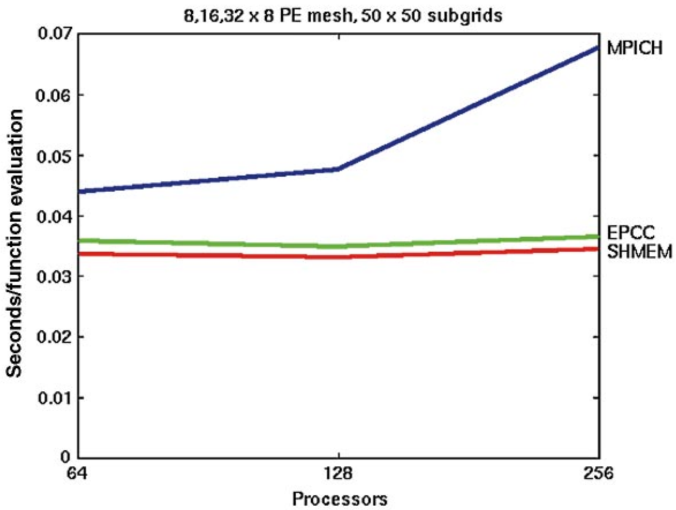
A parallel solver for ordinary differential equation systems, called PVODE, builds upon CVODE by way of parallelization of the module of vector kernels. PVODE contains two of the original four method options, namely the nonstiff method and the stiff method with iterative solution of the linear systems. (A parallel band solver will be added later.)

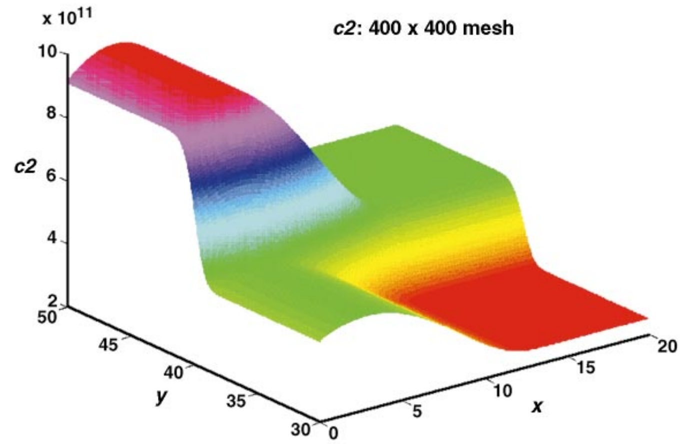
A critical feature in most applications is the preconditioning of the linear systems. Leveraging from other projects, we plan to develop parallel preconditioner modules based on banded and sparse Jacobian matrix approximations, including automatic generation of required matrix data. These modules will address large classes of problems characterized by broad structural features. These parallel solvers and preconditioner modules, combined with automatic generation of the required Jacobian data, will constitute a powerful tool for a wide variety of simulations.

Building on other widely used LLNL software, we plan to develop analogous parallel versions of two other solvers—one for nonlinear equation systems and one for differential-algebraic systems. Our modular design permits this expanded set of solvers to share PVODE's linear solver modules and vector kernels. In contrast to other approaches, the great power of our underlying methods is retained in the parallel versions.

ODE Solver Developed

Our first implementation of PVODE was written for the Cray-T3D machine (256 processors) with its shared memory (SHMEM) programming model. By design, in the user interface to PVODE, the passing





Tests of PVODE on the Cray-T3D, with problem sizes up to 1.28 million dependent variables, show competitive and scalable performance with one implementation of the MPI (Message Passing Interface) library.

of data specific to the machine environment is isolated and minimized, while user calls for machine-independent functions are unchanged from the sequential version. This version also supports a usage mode where PVODE is run on a proper subset of the user's T3D partition.

We have also developed a version of PVODE based on the Message Passing Interface (MPI) system. As before, we needed to rewrite only the vector module in CVODE. We first implemented this version on the IBM-SP2 at Argonne (128 processors) and then moved it to LLNL's Cray-T3D after MPI was installed on that machine.

Testing of both versions has demonstrated proof of the basic design principle, whereby extensions can be isolated to the module of vector kernels. The re-entrant design allows two or more instances of PVODE to be run in parallel. The widespread availability of MPI makes this version of PVODE highly portable. Further, we have found the MPI system to be much easier to work with than the Cray-T3D SHMEM system, because MPI operates at a higher linguistic level.

The modular design of PVODE includes the linear system solvers as separate general-purpose modules. In fact, the parallel version of the preconditioned Krylov algorithm has been used as the linear system solver in a nonlinear steady-state plasma fluid simulation.

The analogous solver for nonlinear algebraic systems is currently being written.

Scalable Performance and User Tools

For each version of PVODE, we ran tests of the vector module alone and then tested the integrator using a series of simple ODE problems. The MPI version of PVODE was tested with both the original Chameleon (MPICH) implementation of MPI and the later Edinburgh (EPCC) implementation.

We ran comparison tests on the Cray-T3D with three versions of PVODE on a test problem with sizes up to 1.28 million dependent variables. The test problem is a pair of two-dimensional kinetics advection–diffusion equations, discretized in space. The plot above shows scaled total cost (cost per function evaluation) as a function of the number of processors, with a fixed subgrid size on each processor. We found that the EPCC version of MPI was quite competitive with the Cray SHMEM version and scaled well with problem size, while the MPICH version did not.

An important part of this project is the development of example programs, or user templates, for the solvers. As a byproduct of the testing, several such example programs have been written for PVODE. Separately, we wrote a set of interface routines between CVODE and FORTRAN to enable users with FORTRAN problem-defining code modules to use CVODE and, eventually, PVODE.

Planned Applications

We are working on applications of this software to two-dimensional tokamak edge-plasma models developed in LLNL's Magnetic Fusion Energy Division. The MFE community's primary tokamak edge model, UEDGE, now uses three of our solvers on sequential machines, and a parallel version of it is planned. Software from this project will enable that code to use problem sizes sufficient to resolve boundary features and impurity effects that are not adequately resolved now. We expect to see numerous other applications of our parallel solvers in a variety of simulations.

For more information about the PVODE project, contact Alan Hindmarsh, 510-422-4276, alanh@llnl.gov.

Work performed under the auspices of the U.S. DOE by LLNL under contract No. W-7405-ENG-48